SYSTEM AND METHOD FOR DIGITAL MEDIA SERVER LOAD BALANCING

FIELD OF THE INVENTION

[0001] This invention relates to the field of load balancing.

BACKGROUND OF THE INVENTION

[0002] Load balancing techniques exist to ensure that individual servers in multi-server systems do not become overloaded and that services retain high availability. Load balancing is especially important where it is difficult to predict the number and timing of requests that will require processing.

[0003] Most current load-balancing schemes employ simple parameters to distribute network traffic across a group of servers. These parameters are usually limited to load amount (measured by the number of received requests), server "health" or hardware status (measured by processor temperature or functioning random access memory), and server availability.

[0004] One common load-balancing architecture employs a supervisor/subordinate approach. In this architecture, a control hierarchy of devices is established in a load-balancing domain. Each server in the system is assigned to a load-balancing group that includes a central device for monitoring the status of servers in its group. The supervisor acts as the gatekeeper for requests entering the group and delegates each request to an appropriate server based on the server's relative status to that of other servers in the group.

[0005] One negative aspect of this approach is that it introduces a single point of failure into the load-balancing process. If the supervisor goes offline for any reason, incoming requests cannot be serviced. To ameliorate this problem, some load-balancing schemes employ a secondary supervisor to handle requests when the primary supervisor is unavailable. A secondary supervisor, however, introduces extra cost in terms of physical equipment and administration.

[0006] One of the earliest forms of load balancing, popular in the early 1990's, is commonly referred to as domain name service (DNS) round robin. This load-balancing scheme, described in connection with Fig. 1, represents an extension of the standard domain name resolution technique primarily used by Internet Web servers experiencing extremely high usage.

[0007] As shown in Fig. 1, in step 110, a client requests data from a DNS server. In step 120, the domain name server resolves the requested server name into a series of server addresses. Each address in the series corresponds to a server belonging to a single load-balancing group. Each server in the group is provided with a copy of all data to be served, so that each server replicates data stored by every other server in the group.

[0008] In step 130, the domain name server assigns new requests by stepping through the list of server addresses, resulting in a crude and unpredictable load distribution for servers in the load-balancing group. Moreover, if the number of requests overloads the domain name server or if the server selected to service the request is at capacity, the service is ungracefully denied. In addition, if the selected server is at capacity, the new request routed by the domain name server may bring the server down.

[0009] Another major problem with DNS round robin is that the domain name server has no knowledge of server availability within the load-balancing group. If a server in the group is down, DNS round robin will nevertheless direct traffic to it.

[0010] In the mid 1990's, second generation load-balancing solutions were released. These solutions employed a dedicated load balance director (LBD), such as Cisco Systems' LocalDirector. The director improves the DNS round robin load-balancing scheme by periodically testing the network port connections of each server in its group and directing responses to responsive servers. One such second generation solution is discussed in "Load Balancing: A Multifaceted Solution for Improving Server Availability" (1998 Cisco Systems, Inc., <http://www.cisco.com/warp/public/cc/pd/cxsr/400/tech/lobal_wp.htm>) which is hereby incorporated by reference.

[0011] A third generation of load-balancing solutions included robust, dedicated load balancing and network management devices, such as the BIG-IP™ from F5 NETWORKS.™ These devices improve server availability by monitoring server health via management protocols such as Simple Network Management Protocol (SNMP). Perhaps the biggest improvement of this generation is the ability to direct traffic based on requested content type instead of just load. For example, requests ending in ".http" are directed to Web servers, ".ftp" to file download servers, and ".ram" to REALNETWORKS'™ streaming servers. This feature enables network managers to create multiple load-balancing groups dedicated to specific content types.

[0012] Although the aforementioned load-balancing techniques are often adequate for managing multi-server systems that serve Web pages, file downloads, databases, and email, they still leave room for significant improvement. Moreover, such load-balancing schemes do not perform well in systems that serve broadcast-quality digital content, which is both time sensitive and bandwidth intensive.

SUMMARY OF THE INVENTION

[0013] A system and method for load balancing a plurality of servers is disclosed. In a preferred embodiment, a plurality of servers in a video-on-demand or other multi-server system are divided into one or more load-balancing groups. Each server preferably maintains state information concerning other servers in its load-balancing group including information concerning content maintained and served by each server in the group. Changes in a server's content status or other state information are preferably proactively delivered to other servers in the group. Thus, for example, to maintain a current inventory of assets within a load-balancing group, each server provides notification to other servers in its group when an asset that it maintains is added, removed, or modified.

[0014] When a content request is received by any server in a load-balancing group, it evaluates the request in accordance with a specified algorithm to determine whether it should deliver the requested content itself or redirect the request to another server in its group. In a preferred embodiment, this determination is a function of information in the server's state table.

[0015] The present system and method provide several benefits. First, because they employ a peer-based balancing methodology in which each server can respond to or redirect client requests, the present system and method do not present a single point of failure, as do those schemes that utilize a single load-balancing director. Second, because the present system and method proactively distribute state information within each group, each server is made aware of the current status of every server in its group prior to a client request. Consequently, when a request for content is received, it may be rapidly directed to the appropriate server without waiting for polled status results from other servers in the group. Moreover, in some preferred embodiments, the present system and method defines parameters concerning the capability of each server such as extended memory, inline adaptable cache, or other unique storage attributes, thus permitting sophisticated load-

balancing algorithms that take account of multiple factors that may affect the ultimate ability of the system to most efficiently respond to client requests. Furthermore, in some preferred embodiments, the present system and method considers other media asset parameters such as whether an asset is a "new release" to help anticipate demand for the asset.

[0016] In one aspect, the present invention is directed to a method for selecting a server from a plurality of servers to service a request for content, comprising: designating a director from the plurality of servers to receive the request, wherein the designation is made on a request-by-request basis; and allocating to the director the task of selecting a server to service the request from the plurality of servers, said server having stored thereon the content, the director using a state table comprising parametric information for servers in the plurality of servers, wherein said parametric information comprises information identifying assets maintained on each server in the plurality of servers.

[0017] In another aspect of the present invention, the step of designating comprises designating the director in a round-robin fashion.

[0018] In another aspect of the present invention, the step of designating comprises designating the director on the basis of lowest load.

[0019] In another aspect of the present invention, the step of selecting further comprises selecting the director if the content is present on the director.

[0020] In another aspect of the present invention, said parametric information further comprises functional state and current load of each server.

[0021] In another aspect of the present invention, said parametric information further comprises whether each server comprises extended memory.

[0022] In another aspect of the present invention, said parametric information further comprises whether each server comprises an inline adaptable cache.

[0023] In another aspect of the present invention, said parametric information further comprises whether each asset is a new release.

[0024] In another aspect of the present invention, the method further comprises rejecting the request if the content is not present on any of the plurality of servers.

[0025] In another aspect of the present invention, the method further comprises forwarding the request to the selected server.

[0026] In another aspect of the present invention, The method further comprises redirecting the request to the selected server.

[0027] In another aspect of the present invention, the step of selecting further comprises: calculating a load factor for each server in the plurality of servers having the content; identifying as available servers one or more servers whose parameters are below threshold limits; selecting a server from the available servers having the lowest load factor; and otherwise selecting a server having the lowest load factor from the plurality of servers having the content.

[0028] In another aspect, the present invention is directed to a server for directing a request for content among a plurality of servers comprising: a state table comprising parametric information for each server in the plurality of servers, said parametric information comprising information identifying assets maintained on the plurality of servers; and a communication component for sending changes to the state table to the plurality of servers.

[0029] In another aspect of the present invention, the server is a member of a load-balancing group, and the communication component sends changes to servers in the load-balancing group.

[0030] In another aspect of the present invention, the server further comprises a redirection means for acknowledging the client request and identifying one of the plurality of servers where the requested asset is stored.

[0031] In another aspect of the present invention, the server further comprises a forwarding means for sending the client request to one of the plurality of servers where the requested asset is stored.

[0032] In another aspect of the present invention, said parametric information further comprises functional state and current load of each server.

[0033] In another aspect of the present invention, said parametric information further comprises whether each server comprises extended memory.

[0034] In another aspect of the present invention, said parametric information further comprises whether each server comprises an inline adaptable cache.

[0035] In another aspect of the present invention, said parametric information further comprises whether each asset is a new release.

BRIEF DESCRIPTION OF THE DRAWINGS

[0036] Fig. 1 is a flowchart illustrating a DNS round robin load-balancing scheme in the prior art;

[0037] Fig. 2 is a block diagram illustrating an exemplary designation of servers into load-balancing groups;

[0038] Fig. 3 is a diagram illustrating exemplary state tables in a preferred embodiment of the present system and method;

[0039] Fig. 4 is a flowchart illustrating a preferred embodiment of a process for updating state tables;

[0040] Fig. 5A and 5B are flowcharts illustrating a preferred embodiment of the present system and method for load-balancing in a multi-server system;

[0041] Figs. 5C and 5D are block diagrams illustrating the communication paths for forwarding and redirecting client content requests in a preferred embodiment of the present system and method;

[0042] Fig. 6 is a flowchart illustrating a preferred embodiment of the present system and method for load-balancing in a multi-server system with replicated content;

[0043] Fig. 7 is a flowchart illustrating a preferred load-balancing algorithm; and

[0044] Fig. 8 is a diagram illustrating an exemplary state table in a preferred embodiment of the present system and method.

DESCRIPTION OF THE PREFERRED EMBODIMENTS

[0045] Fig. 2 shows an exemplary digital media delivery system that includes six digital media servers A-F which provide content to a plurality of clients via a network. In the

exemplary embodiment of Fig. 2, the six servers are divided into two load-balancing groups (LBG) 210, 220. Servers A-C are designated as belonging to load-balancing group 1 (210) and servers D-F are designated as belonging to load-balancing group 2 (220). Load-balancing groups 210, 220 may or may not be geographically diverse, but are logical groupings that designate which servers will share state information in a preferred embodiment of the present system and method, as described below.

[0046] Each server A-F preferably maintains state information concerning one or more parameters associated with each server in its group. Accordingly, each of servers A-C preferably maintains such state information for servers A-C and each of servers D-F preferably maintains such state information for servers D-F.

[0047] One preferred embodiment for maintaining state information concerning servers in a load-balancing group is shown in Fig. 3. As shown in Fig. 3, a first table 310 preferably comprises a row for each server in load-balancing group 210 and a second table 320 preferably comprises a row for each server in load-balancing group 220. Each table 310, 320 also preferably comprises a plurality of columns for storing state information concerning a plurality of parameters that may be considered in load-balancing determinations, as described below. Such parameters may include, without limitation, Moving Picture Experts Group (MPEG) bandwidth limits, total external storage capacity, addressable service groups (i.e., customer set-top boxes), current number of streams, system central processing unit (CPU) utilization, user CPU utilization, CPU idle, outgoing stream bandwidth, incoming bandwidth for network interfaces, streaming server state, system uptime, time-averaged load, server temperature, memory usage, total available cache, cache used, total external storage remaining, total external storage used, MPEG stream count limit, status of network connections, and status of power supplies, fans, and storage devices.

[0048] In a preferred embodiment, one or more of the stored parameters relate to the asset inventory of each server. The state table may also store other media asset parameters such as whether the asset is a "new release" to help anticipate demand for the asset. The state table additionally may contain parameters concerning the capability of each server such as whether it comprises extended memory or an inline adaptable cache (such as that described in U.S. patent application serial No. ____, entitled _____, filed _____, (identified by Pennie & Edmonds docket No. 11055-013) which is hereby incorporated by reference in its entirety for each of its teachings and embodiments), or other unique storage attributes.

[0049] In a preferred embodiment, threshold limits may be specified for one or more of the stored parameters that represent an unacceptable condition. Use of these threshold limits in selecting a server to deliver requested content is described in more detail below.

[0050] A preferred embodiment for updating state tables 310, 320 at each server is illustrated in Fig. 4. As shown in Fig. 4, in step 410, all servers in a load-balancing group initially have identical state tables. In step 420, a parameter of server A is modified (e.g., an asset is copied onto server A), thus changing server A's state. In step 430, server A updates its own state table, and pushes the state change information to all other servers in its load-balancing group. In a preferred embodiment, this state change information is transmitted concurrently to all other servers in the group via, for example, a multicast, broadcast, or other one-to-many communications mechanism. In step 440, the other servers update their state tables with the state change information, and the state tables of all servers are again synchronized. In addition, each server is preferably adapted to add or remove parameter columns from its state tables, so that load-balancing algorithms applied by the servers may change over time and take account of different combinations of parameters.

[0051] Fig. 5A illustrates a preferred embodiment for responding to a request for content from, for example, a client set-top box. In step 510, the content request is received by a business management system (BMS) of the digital media delivery system. The business management system is preferably adapted to provide billing, authentication, conditional access, programming guide information, and other functions to client set-top boxes.

[0052] In step 512, the business management system authenticates the client and bills the client for the request. In step 514, the business management system designates one of the media servers of the digital media delivery system to act as a director for this request. The role of the director is to select an appropriate server to deliver the requested content to the client, as described below. In a preferred embodiment, the director may be selected by the business management system on a rotating basis. In an alternative preferred embodiment, the director may be selected on the basis of server load (i.e., the server with lowest current load is designated to act as director for the request).

[0053] In step 516, the server designated to act as director for this request selects a server from its load-balancing group to deliver the requested content to the client. As described below, this server may be the director itself or another server in its group.

Preferred embodiments for making this selection are described below in connection with Figs. 5B and 6.

[0054] In step 518, the server selected to deliver the content sets up a streaming session and notifies the business management system that it is ready to stream the requested content to the client. In step 520, the business management system directs the client to the IP address of the selected server and delivery of the requested content is commenced.

[0055] In an alternative preferred embodiment, after selecting a server to act as director for a request, the business management system provides the director's IP address directly to the client. In this embodiment, the client contacts the director which selects a server to provide the requested content and then provides that server's IP address to the client when the streaming session is set up.

[0056] One preferred embodiment that may be utilized by a director for selecting a server to deliver requested content is now described in connection with Fig. 5B. As shown in Fig. 5B, in step 530, a content request is received by the server designated by the business management system to act as director for the request. In step 540, the director identifies the requested content and determines whether or not the director has this content available. If the content is available from the director itself (step 550), it designates itself to deliver the requested content (step 555). Otherwise, in step 560, the server examines its state table 310 to see if the content is available from another server in its load-balancing group. If the content is not available in the group, the director rejects the request (step 565). Otherwise, as described above, the director instructs the selected server to set up a streaming session for the client and redirects the client to submit the request to that server when the session is set up (step 570). Two alternatives to step 570 are described respectively in connection with Figs. 5C and 5D. In the first alternative, the director forwards the request to the selected server which directly respond to the client when the streaming session is set up. In the second alternative, the director redirects the client to the selected server and the client directly requests establishment of a streaming session from the selected server.

[0057] This first alternative is illustrated in an exemplary communication block diagram shown in Fig. 5C. More specifically, as shown in Fig. 5C, in step 530, server A receives a request from the client. In step 570, server A concludes that a different server in the group can service the request (server C in this example), and forwards the request to that

server. Server C processes the forwarded request as if it were received directly from the client, and performs steps 540-555 shown in Fig. 5B to deliver the requested content to the client.

[0058] Turning to Fig. 5D which illustrates the second alternative, in step 530, server A receives a request from the client. In step 570, server A concludes that a different server in the group can service the request (server C in this example), and sends an acknowledgment to the client, redirecting the client to that server. The client then retransmits its request to server C (step 530). Server C processes the request, and performs steps 540-555 shown in Fig. 5B to deliver the requested content to the client.

[0059] In a preferred embodiment, content may be replicated on multiple servers in a load-balancing group to satisfy request volumes that may exceed a single server's capacity. Moving or copying content from one server to another in a load-balancing group may also be used as a strategy to further distribute load within the group. Fig. 6 illustrates a preferred system and method for load-balancing content requests in a multi-server system with replicated content.

[0060] As shown in Fig. 6, in step 610, a client makes a request for content. The request is forwarded to a business management system which designates a server as director for this request and forwards the request to the director, as described above. In step 620, the director analyzes the request to identify the requested content. In step 630, the director determines if the requested content is present in the load-balancing group by consulting its state table. If the content is not present in the load-balancing group, the director rejects the request (step 640).

[0061] In an alternative embodiment, the director may forward the request to a server in another load-balancing group. This alternative, however, suffers from significant drawbacks, because the director in the present embodiment has no knowledge whether the content is present in the other load-balancing groups, and a poor level of service may result depending upon the ability of a second load-balancing group to provide the content. To overcome this drawback, each server may be provided with additional state tables with information concerning servers in other load-balancing groups. Alternatively, all servers in the system may be designated as belonging to a single load-balancing group. These

alternatives, however, present their own disadvantages including increased overhead to update and maintain state tables.

[0062] Returning to Fig. 6, if the content is available in the load-balancing group, the server examines its state table to identify those servers in the group that have the requested content (step 650). In step 660, the server applies a load-balancing algorithm to choose a server in its group to supply the requested content. One preferred embodiment of such an algorithm is described in more detail below. In step 670, the client request is redirected or forwarded to the selected server as described above in connection with Figs. 5B-5D. In step 680, the selected server delivers the requested content to the client. In step 690, the selected server updates its state table to reflect corresponding changes in its load and other parameters and communicates these state-table changes to the other servers in its load-balancing group.

[0063] A preferred embodiment of a load-balancing algorithm for selecting a server to deliver requested content is illustrated in Fig. 7. As shown in Fig. 7, in step 710, the director examines its state table to identify all servers in its load-balancing group that have the requested content and are operational (referred to hereafter as target servers).

[0064] In step 720, the server calculates a load factor for each of the target servers from a weighted sum of parameters indicative of load. In a preferred embodiment, the parameters used to calculate the load factor for each server are: incoming streaming bandwidth, outgoing streaming bandwidth, total storage usage, memory usage, and CPU utilization.

[0065] In step 730, the server determines whether any target servers have exceeded a parameter threshold limit. For example, a target server may have an abundance of outgoing streaming bandwidth available, but the server's CPU utilization parameter may be very high and exceed the threshold limit established for that parameter. This target server would therefore not be a preferred choice to serve the requested content. As used herein, the term available servers refers to target servers that have not exceeded any threshold limits.

[0066] In step 740, the server determines if there are any available servers. If so, in step 750, the server chooses the available server having the lowest load factor to deliver the requested content. If not, then in step 760, the server chooses the target server having the lowest load factor from all target servers.

[0067] Fig. 8 shows an exemplary state table suitable for illustrating the above-described process of Fig. 7. For purposes of the present example, it is assumed that the three listed servers A-C are members of load-balancing group 1 and that each maintains the exemplary state table of Fig. 8. It is further assumed that a client places a request to view the asset "Dare Devil," a feature-length film.

[0068] The director, assume server B, examines its state table and determines that the content for "Dare Devil" is stored on servers A and C. Since servers A and C are up, they are the target servers.

[0069] Server B then calculates the load factor for each of the target servers. The load factor is preferably defined to be a weighted average of parameters. For the purpose of this example, it is assumed that the bandwidth capacity, both incoming and outgoing, is 500, and the load factor is expressed as an average of each parameter, measured in percent capacity. Thus, server B would determine the load factor of server A as (4700/500+27500/500+34+37+40)/5 % = 35%, and the load factor of server C as (1300/500+39600/500+56+60+64)/5 % = 52.4%.

[0070] Next, server B determines whether both servers are available. For the purpose of this example, it is assumed that the threshold limit set for each parameter on each server is 75%. Since no threshold limits are exceeded by any target server, servers A and C are both available servers. Since there is at least one available server, server B chooses the server with the lowest load factor, namely server A.

[0071] As server A starts supplying the "Dare Devil" content, it updates its state-table parameters to reflect this fact (e.g., overall load, bandwidth, etc.). Server A preferably broadcasts these changes to all other servers in its load-balancing group, as described above.

[0072] While the invention has been described in conjunction with specific embodiments, it is evident that numerous alternatives, modifications, and variations will be apparent to persons skilled in the art in light of the foregoing description.